Data and Process Modelling

9. Formal Analysis of Process Control-Flow with Petri-Nets

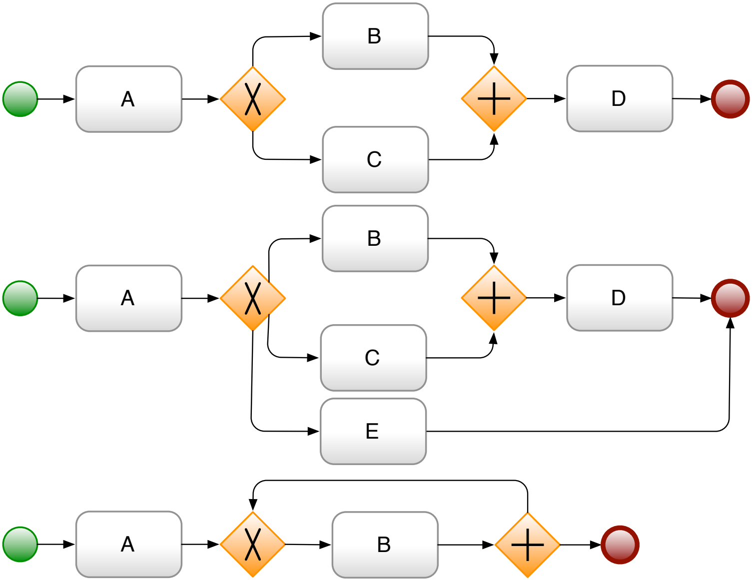
Marco Montali

KRDB Research Centre for Knowledge and Data
Faculty of Computer Science
Free University of Bozen-Bolzano

A.Y. 2014/2015

Correctness of Designed Models

Are these models correct?



Petri Nets

- Introduced by Carl Adam Petri in his PhD thesis (1962).
- Original intention: mathematical description of *chemical processes*.
- Extensively applied to model *concurrent systems* (e.g., distributed systems) and analyse their properties.
  - General properties (e.g., termination, absence of deadlocks) vs particular properties (e.g., reachability of a given desired situation).
- Then extensively investigated to tackle the control-flow of BPs and (web) services behavior.
- Minimal notation: places, transitions, arcs (with multiplicities).
- Several extensions of basic Petri nets, with increasing level of complexity.
  - Time, resources, data (colored Petri nets), hierarchies (process decomposition), open nets (service interaction),...
- Different reasonable restrictions on the structure of the net, with positive impact on complexity.
  - In the BPM context: choice-free nets, workflow nets.

Petri Net

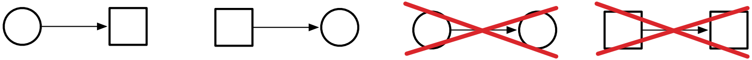
A bipartite oriented graph with two kinds of nodes (places, transitions) and arcs annotated with weights (multiplicities).

Petri net

A *Petri net* is a tuple (P, T, F, W), where:

- P is a finite set of places;
- T is a finite set of transitions, with $P \cap T = \emptyset$;
- $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs forming a flow relation;
- $W : F \longrightarrow \mathbb{N} \setminus \{0\}$ is an (arc) weight function.

- Graphical notation: places $= \bigcirc$, transitions $= \square / []$, arcs $= \rightarrow$.
- Arc types:



Preset and Postset

Multi-set

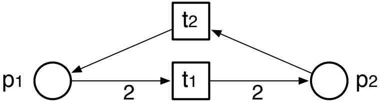
Given a set S, $\mathbb{B}(S) : S \longrightarrow \mathbb{N}$ is the set of multi-sets over S.
$X \in \mathbb{B}(S)$ is a multi-set where, for each $a \in S$, $X(a)$ denotes the number of times a is included in X.

Multisets are represented using $[\cdots]$, and for compactness elements are represented using "power notation" ($a^{X(a)}$): $[a, a, a, b, c, b] = [a^3, b^2, c]$.

Preset/postset

Given a Petri net (P, T, F, W) and $a \in P \cup T$:

- $\bullet a = \left[x^{W(x,a)} \mid W(x, a) \text{ is defined and } (x, a) \in F \right]$;
- $a \bullet = \left[x^{W(a,y)} \mid W(a, y) \text{ is defined and } (a, y) \in F \right]$.



$\bullet p_1 = [t_2]$
$p_1 \bullet = [t_1^2]$

$\bullet t_2 = [p_2]$
$t_2 \bullet = [p_1]$

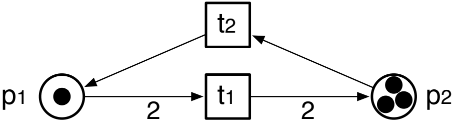
Tokens and Marking

We populate a Petri net with tokens.

Marking

A marking M of a Petri net (P, T, F, W) is a multi-set over P: $M \in \mathbb{B}(P)$.

The marking identifies how many tokens are currently present in each place of the net.



$M_0 = [p_1^1, p_2^3]$.

Compact notation: $M_0 = (1, 3)$.

Firing Rule

Given a marking, the firing rule determines whether a transition can fire (i.e., be executed) and what is the resulting new marking.

Firing rule

Given a Petri net $N = (P, T, F, W)$ and a marking $M \in \mathbb{B}(P)$:

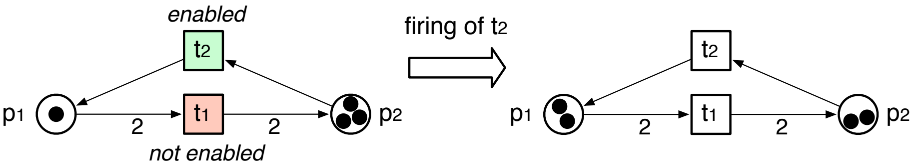
- a transition $t \in T$ is enabled, denoted $(N, M)[t\rangle$, if and only if $M \geq \bullet t$;
- an enabled transition $t \in T$ can fire leading to marking $M' \in \mathbb{B}(P)$, denoted $(N, M)[t\rangle(N, M')$, if and only if $M' = (M - \bullet t) + t\bullet$.

The notions of sub-multi-set $\geq$, multi-set difference $-$ and multi-set sum $+$ are defined following the intuition (component by component).

Firing Rule - Intuition

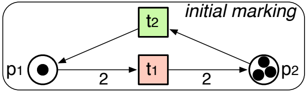
The firing of a transition determines an execution step of the net.

- A transition can fire if there are sufficiently many tokens in each of the input places (as required by the arcs' weights).
- The result is obtained by removing the necessary tokens from each input place, and producing the necessary tokens in each output place (as required by the arcs' weights).



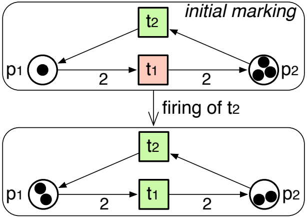
Firing Rule - Non-Determinism

- Starting from an initial marking, a sequence of firings determines an execution of the net.
- At every step, in general there are many enabled transitions.
- One of them is chosen non-deterministically: token game.



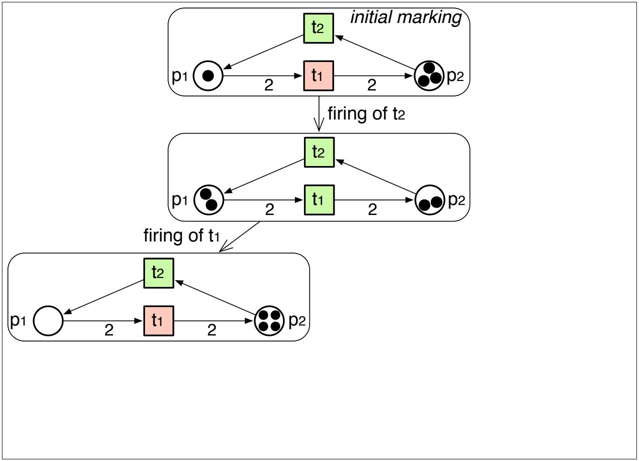
Firing Rule - Non-Determinism

- Starting from an initial marking, a sequence of firings determines an execution of the net.
- At every step, in general there are many enabled transitions.
- One of them is chosen non-deterministically: token game.



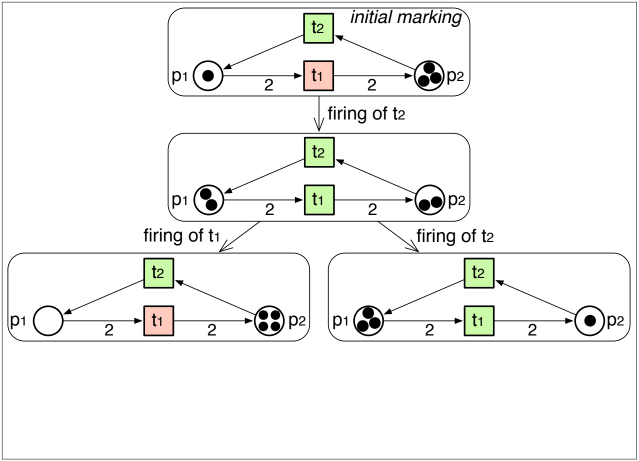
Firing Rule - Non-Determinism

- Starting from an initial marking, a sequence of firings determines an execution of the net.
- At every step, in general there are many enabled transitions.
- One of them is chosen non-deterministically: token game.



Firing Rule - Non-Determinism

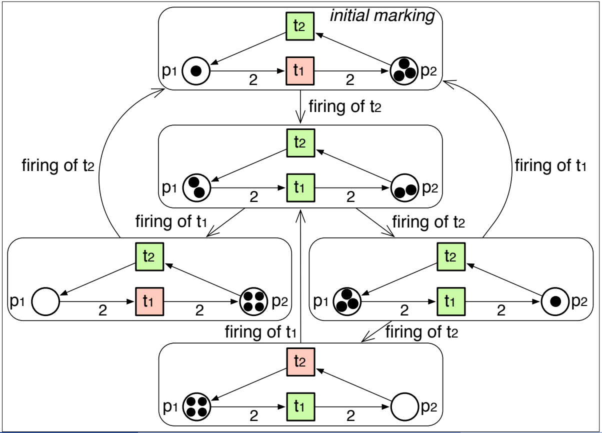
- Starting from an initial marking, a sequence of firings determines an execution of the net.
- At every step, in general there are many enabled transitions.
- One of them is chosen non-deterministically: token game.



Reachability graph

By iterating for each possible enabled transition in each produced marking, a transition system is obtained that represents all the possible executions.

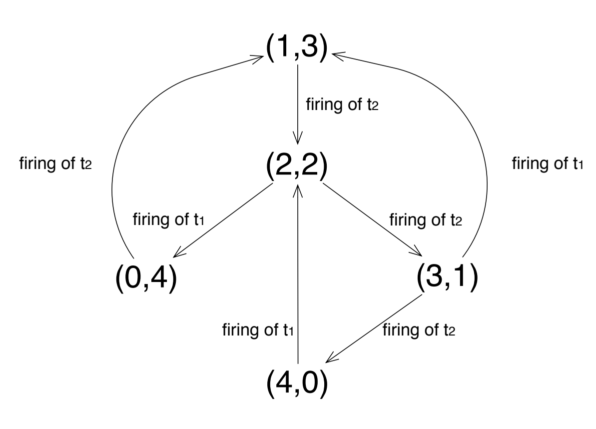
- The transition system is in general *infinite-state*.
- The transition system includes all the *reachable* markings, and is therefore called reachability graph.



Reachability graph

By iterating for each possible enabled transition in each produced marking, a transition system is obtained that represents all the possible executions.

- The transition system is in general *infinite-state*.
- The transition system includes all the *reachable* markings, and is therefore called reachability graph.

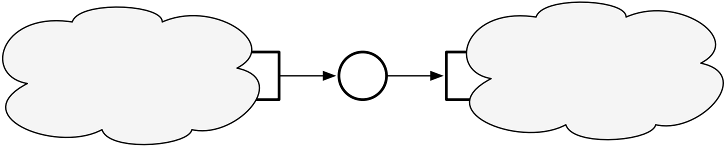


Petri Nets and Business Processes

Petri nets are a natural formalism to represent the *control-flow* of BPs.

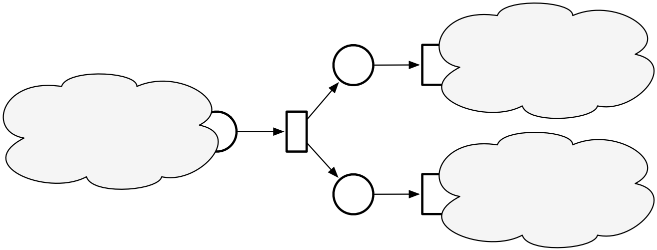
Petri Net Concept	BP Concept
Place	State
Transition	Atomic activity/event in the activity life-cycle
Token	Object manipulated by a process instance (patient, order, item, ...)
Marking	Snapshot of a process instance
Initial marking	Initial state of a process instance
Enabled transition	Executable activity/event
Firing	Execution step of the process
Reachability graph	Transition system representing all possible executions of the process

Petri Nets and Workflow Patterns: Sequence

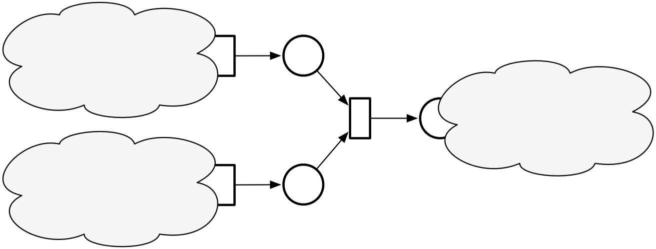


Petri Nets and Workflow Patterns: And-Split/Join

And-split

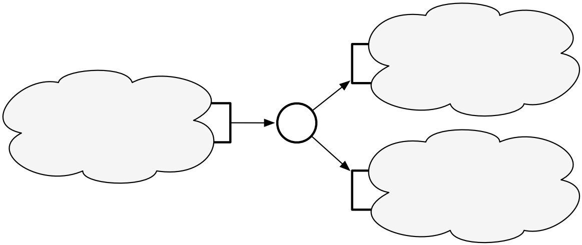


And-join

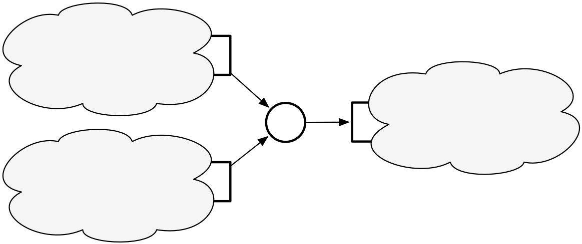


Petri Nets and Workflow Patterns: Xor-Split/Join

Xor-split

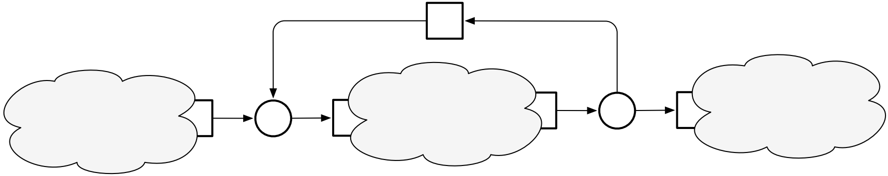


Xor-join

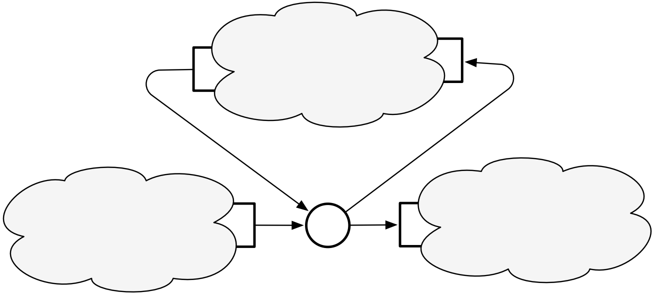


Petri Nets and Workflow Patterns: Arbitrary Loops

1 or more times

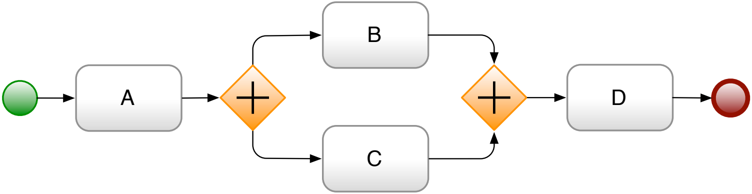


0 or more times



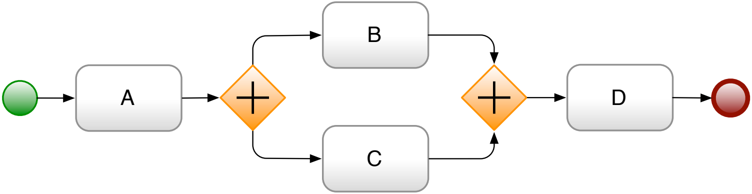
Example

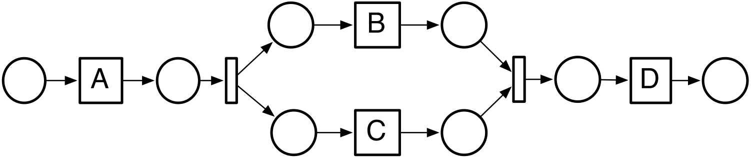
Translate the following BPMN process diagram into a corresponding Petri net, and draw the reachability graph starting from a marking where a single token is put into the starting place.



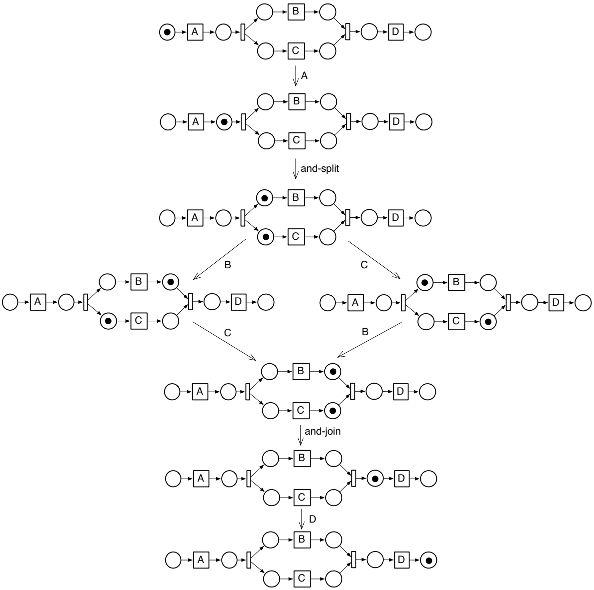
Example

Translate the following BPMN process diagram into a corresponding Petri net, and draw the reachability graph starting from a marking where a single token is put into the starting place.





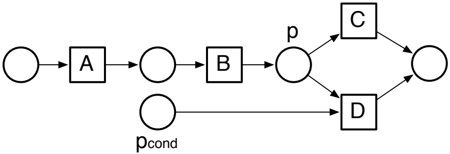
Example - Reachability Graph



Interleaving semantics for parallelism: parallelism between B and C represented as the sequence B,C or the sequence C,B.

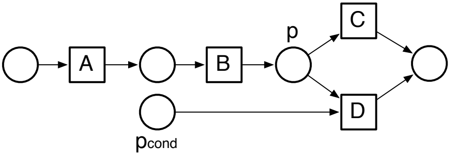
Free-Choice Nets

Consider this Petri net:



Free-Choice Nets

Consider this Petri net:



The x-or choice modeled in p is *conditioned* by place p_{cond}:

- C can be always chosen;
- D can be chosen only if there is a token in p_{cond}.

The choice is *not free*.

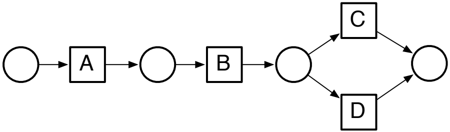
In BPs, choices are instead typically *free*: they depends only on the data associated to the x-or place (p), or on the external decision of responsible resources (deferred choice).

Free-Choice Net

Free-choice net

A Petri net (P, T, F, W) is *free-choice* if, for each $f = (p, t) \in F$:

- $|p \bullet| = 1$ (f is the unique outgoing arc from p), or
- $|\bullet t| = 1$ (f is the unique incoming arc to t).

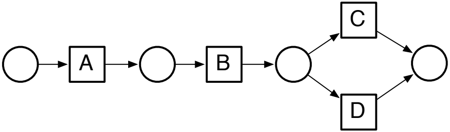


Free-Choice Net

Free-choice net

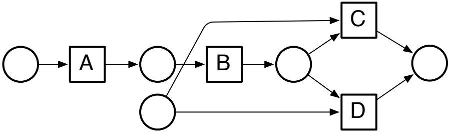
A Petri net (P, T, F, W) is *free-choice* if, for each $f = (p, t) \in F$:

- $|p \bullet| = 1$ (f is the unique outgoing arc from p), or
- $|\bullet t| = 1$ (f is the unique incoming arc to t).



(Extended) free-choice net

A Petri net (P, T, F, W) is *(extended) free-choice* if, for each $p_1, p_2 \in P$, either $p_1 \bullet \cap p_2 \bullet = \emptyset$, or $p_1 \bullet = p_2 \bullet$.



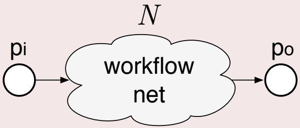
Workflow Net

BPs typically have a starting point and a termination point (explicit end).

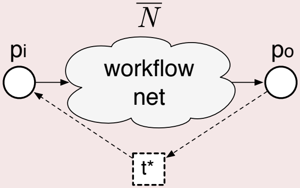
Workflow net

A Petri net $N = (P, T, F, W)$ is a *workflow net* if

- There are two special places in P:
  - an input place $p_i \in P$ such that $\bullet p_i = \emptyset$;
  - an output place $p_o \in P$ such that $p_o \bullet = \emptyset$.



- By adding a transition t^* from p_i to p_o, the resulting Petri net $\overline{N}$ is strongly connected: every pair of nodes (transition of places) of N are connected via a direct path.



Some Fundamental Properties of Petri Nets

Given a Petri net N and an initial marking M:

- (N, M) is terminating iff there exists $k \in \mathbb{N}$ such that any firing sequence from M has a length $\leq$ k.
- (N, M) is deadlock-free iff for every marking M' reachable from M there exists an enabled transition in M'.
- Place p of N is k-bounded in (N, M) iff for every marking M' reachable from M, M' assigns to p at most k tokens.
- (N, M) is k-bounded iff every place of N is k-bounded in (N, M).
- (N, M) is safe iff (N, M) is 1-bounded.
- Transition t of N is live in (N, M) iff for every marking M' reachable from M, there exists a marking M'' reachable from M' such that t is enabled in M''.
- (N, M) is live iff every transition of N is live in (N, M).

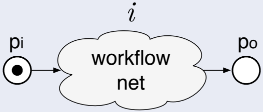
Workflow Nets and Special Markings

Workflow nets have two interesting markings.

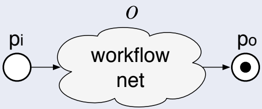
Input/output state

Given a workflow net N:

- The input state i is a marking that assigns only one token to the input place p_i of N.



- The output state o is a marking that assigns only one token to the output place p_o of N.



Workflow Nets and the Soundness Property

Soundness

A workflow net N is *sound* if and only if:

1. $(\overline{N}, i)$ is deadlock-free: starting from the initial marking the only situation in which no transition is enabled is only o.
2. Starting from the input state i, the output state is always reachable: for every marking M reachable from i, there exists a firing sequence leading to o.
3. The output place p_o is marked only in a clean way by o: whenever a token is put in place p_o, all the other places are empty.

Theorem (van der Aalst, 1997)

A workflow net N is sound if and only if $\overline{N}$ is live and bounded.

Theorem (van der Aalst, 1997)

For a free-choice workflow net it is possible to decide soundness in polynomial time.

Back to the Reachability Graph

Construction algorithm

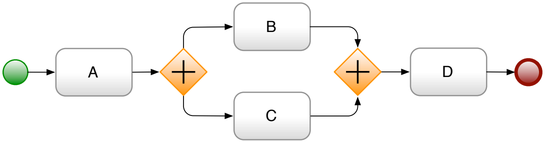
Given a Petri net N and an initial marking M_0:

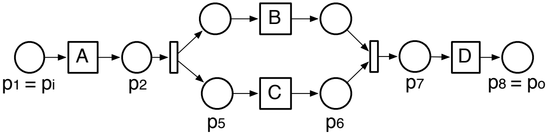
1. Label M_0 as the *root* and initialize set $New = \{M_0\}$.
2. While $New \neq \emptyset$:
   - 2.1 Select marking M from New.
   - 2.2 While there exists an enabled transition t at M:
     - 2.2.1 Obtain the marking M' that results from firing t at M.
     - 2.2.2 If M' does not appear in the graph add it to the graph and insert M' into set New.
     - 2.2.3 Draw an arc with label t between M and M'.
   - 2.3 Remove M from New.

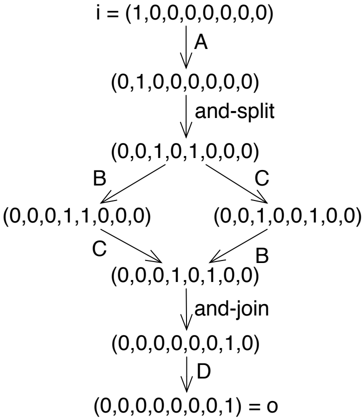
Question

Does this algorithm always terminate?

Example - Sound Process



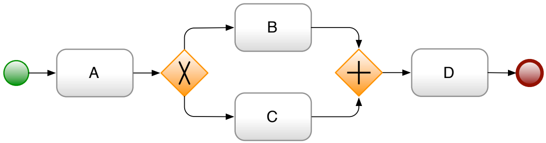


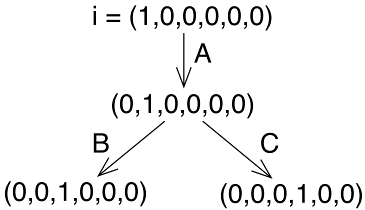


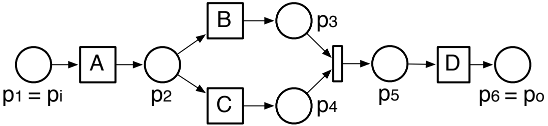
Why? Check reachability graph wrt the three properties for soundness:

1. OK! The only reachable marking without outgoing edges (i.e., no enabled transitions) is o.
2. OK! Marking o is reachable from all the other markings.
3. OK! The only reachable marking that puts a "1" in the last position (i.e., that puts a token into p_o) is o.

Example - Unsound, Deadlocking Process



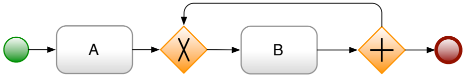


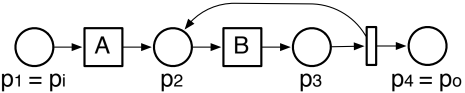


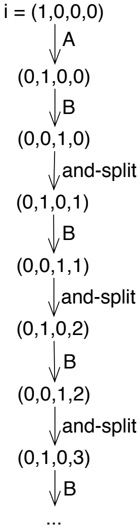
Why?

1. NO! There are two reachable markings different than o for which there is no enabled transition.
2. NO! Marking o is not reachable.
3. OK! No reachable marking exists that puts a token in p_o and at the same time tokens in other places.

Example - Unsound, Unbounded Process







Why?

1. OK! All reachable markings have at least one transition enabled (in fact, exactly one).
2. NO! Marking o is not reachable.
3. NO! There are reachable markings that associate a token to p_o and at the same time tokens to other places, such as $(0,1,0,1)$ and $(0,1,0,2)$.

N.B.: Infinite reachability graph!!!

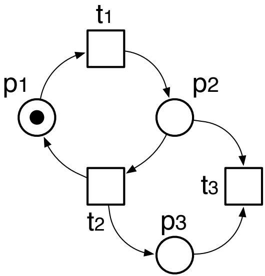
The Problem of Boundedness

The previous example shows that we cannot always construct the reachability graph. The problem arises when the marked net is unbounded.

Question

How to decide boundedness?

Consider the following example:



Fire t_1 and then t_2. What happens?

- We obtain a marking that "includes" the starting one.
- The behavior of a Petri net is monotonic: if a transition is enabled in a marking M, it will be enabled in all those markings that include M.
- We can imagine to "accelerate" the net, by continuing to execute t_1 and t_2.
- The result is that we continue to end up in the same situation, apart from p_3, which continues to accumulate new tokens $\rightsquigarrow$ put ω instead for the actual number.

Abstract Marking

ω denotes that a place is unbounded. Mathematically:

- Now a marking assigns to each place an element from $\mathbb{N} \cup \{\omega\}$.
- We extend the multiset operators accordingly:
  - $\omega \geq \omega$, and $\omega > n$ for every $n \in \mathbb{N}$.
  - An unbounded place will be unbounded forever: $\omega + n = \omega$, $\omega - n = \omega$.

Through "acceleration", we construct a finite abstraction of the reachability graph that exploits ω markings to denote unbounded places.

- Infinite parts of the reachability graph are finitely summarized.

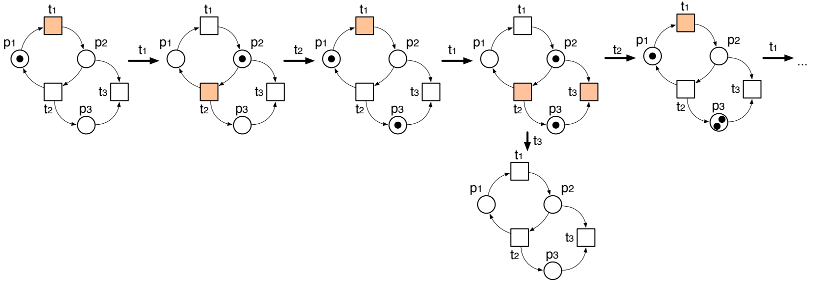
Abstract Marking

ω denotes that a place is unbounded. Mathematically:

- Now a marking assigns to each place an element from $\mathbb{N} \cup \{\omega\}$.
- We extend the multiset operators accordingly:
  - $\omega \geq \omega$, and $\omega > n$ for every $n \in \mathbb{N}$.
  - An unbounded place will be unbounded forever: $\omega + n = \omega$, $\omega - n = \omega$.

Through "acceleration", we construct a finite abstraction of the reachability graph that exploits ω markings to denote unbounded places.

- Infinite parts of the reachability graph are finitely summarized.



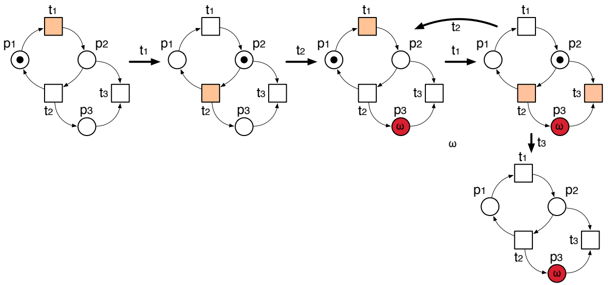
Abstract Marking

ω denotes that a place is unbounded. Mathematically:

- Now a marking assigns to each place an element from $\mathbb{N} \cup \{\omega\}$.
- We extend the multiset operators accordingly:
  - $\omega \geq \omega$, and $\omega > n$ for every $n \in \mathbb{N}$.
  - An unbounded place will be unbounded forever: $\omega + n = \omega$, $\omega - n = \omega$.

Through "acceleration", we construct a finite abstraction of the reachability graph that exploits ω markings to denote unbounded places.

- Infinite parts of the reachability graph are finitely summarized.



Coverability Graph

Construction algorithm

Given a Petri net N and an initial marking M_0:

1. Label M_0 as the *root* and initialize set $New = \{M_0\}$.
2. While $New \neq \emptyset$:
   - 2.1 Select marking M from New.
   - 2.2 While there exists an enabled transition t at M:
     - 2.2.1 Obtain the marking M' that results from firing t at M.
     - 2.2.2 **For every marking $M'' \neq M'$ on a path from M_0 to M': if $M'' \leq M'$, then for every place p s.t. $M'(p) > M''(p)$, set $M'(P) = \omega$.**
     - 2.2.3 If M' does not appear in the graph add it to the graph and insert M' into set New.
     - 2.2.4 Draw an arc with label t between M and M'.
   - 2.3 Remove M from New.

Question

Does this algorithm always terminate?

Reachability vs Coverability Graph

Does the coverability graph faithfully represent the reachability graph?

NO! When we have a marking that assigns ω to place P, then, for any number $n \in \mathbb{N}$, we now that it will be possible to reach a state in which P contains **at least** n tokens.

Observations:

- When ω markings are present, the coverability graph cannot be used to answer *reachability queries*, but only *coverability queries*.
- *Different* Petri nets could have the *same* coverability graph due to the abstraction.
- The *same* Petri net could have *different* coverability graphs due to non-determinism.
- Boundedness is correctly decided by checking whether the coverability graph contains ω markings or not.
- Every run of the Petri net can be executed over the coverability graph, but not the other way around.
- Hence, liveness cannot be correctly decided by checking the coverability graph.
- A transition is *dead* if and only if *it does not appear* in the coverability graph.
- When the marked net is bounded, then the coverability and the reachability graphs coincide.

Cf. examples on the blackboard!

Complete Procedure for Soundness

Given a workflow net N (with input state i)...

1. Construct the coverability graph for $(\overline{N}, i)$.
2. Use the coverability graph to check whether $(\overline{N}, i)$ (and, in turn, (N, i)) is bounded.
3. If not $\rightsquigarrow$ return NO.
4. If so (the coverability graph and the reachability graph coincide):
   4.1 Check whether $(\overline{N}, i)$ is live.
   4.2 If so $\rightsquigarrow$ return YES.
   4.3 If not $\rightsquigarrow$ return NO.

Final Remarks

- Reachability graph can be infinite $\rightarrow$ coverability graph that uses ω-markings to compactly represent the sources of unboundedness.
- State-explosion problem: the coverability graph can be huge $\rightsquigarrow$ **exponential space in the size of the original net**.
- Structural analysis is used to check properties without constructing the coverability graph explicitly.
  - Place invariants, traps, ...